Attorney's Docket No. *081862.P123*
Cisco Disclosure Number T025-WANBU
Express Mail No. EM560888573US

UNITED STATES PATENT APPLICATION

FOR

CALL RECORD MANAGEMENT FOR HIGH CAPACITY SWITCHED VIRTUAL CIRCUITS

Inventors:

Henry Fourie

Snehal Karia

Prepared by:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

12400 Wilshire Boulevard, Seventh Floor

Los Angeles, California 90025-1026

(310) 207-3800

CALL RECORD MANAGEMENT FOR HIGH CAPACITY SWITCHED VIRTUAL CIRCUITS

FIELD OF THE INVENTION

The invention relates generally to communications and networking. More specifically, the invention relates to the usage of resources in networking devices.

BACKGROUND OF THE INVENTION

In connection-oriented networking schemes such as ATM (Asynchronous Transfer Mode), connections or "calls" must be established between one information device such as a computer system or router and another. This call or connection is sometimes referred to as a "virtual circuit" (VC) particularly where a specified data pipe is artificially, through software, segmented into separate data-pathways, each pathway servicing a particular VC. Often a switch acts as an intermediary to direct one or more of these VCs through a particular network node, and thus these calls are collectively referred to as SVCs (Switched Virtual Circuits).

Figure 1 shows an exemplary wide-area networking system serviced by ATM. A wide-area network (WAN) link 120 interconnects a first network 100 with a second network 110. Each network has a plurality of nodes that may each contain switching devices that regulate data traffic to one or more user terminals. Network 100 is shown having nodes 102, 103,

104, 106, and 108, while network 110 is shown having nodes 112, 114, 116, and 118. A first user terminal 105 is connected to node 102 of network 100 while a second user terminal 115 is connected to node 118 of network 110. In order for user terminal 105 and user terminal 115 to communicate with one another, a call must first be established between them. This call may be switched through a plurality of nodes. One possible route for sending data from user terminal 105 to user terminal 115 is for data to go from node 102 to node 106 to node 108 and then across the WAN link to node 112 and node 118 finally reaching user terminal 115. Each node has a controller device (SVC controller) and switch which facilitates the calls through its node. The SVC controller has processing, memory and other resources to interpret, forward and process messages and initiate other messages as appropriate, while the switch ordinarily handles the physical routing of messages among nodes and user terminals.

Similar to PSTN (Public Switched Telephone Network) communications such as telephone calls, the period of SVC call operation for a given call can be split into three distinct phases—establishment (setup), active (data transfer), and disconnect (hang-up). Once a call is established, for example, between user terminal 105 and user terminal 115

across a specified path, a virtual circuit will have been created and the call can proceed into the active phase where data is transferred. Once the data transfer is complete, the call can be disconnected, which will release the virtual circuit. State, signaling, and other information for each call that passes through a network node is memorialized in a "call record" stored in the that node whether that node is the source, an intermediary, or destination node. The call record is updated whenever a change in state or activity in the call is indicated. The call record may be used for functions such as billing, tracing, routing, etc.

Ordinarily the creation, storage, updating and retrieval either in whole or part by the SVC controller device within the node. The capacity (number of supportable connections or calls) of the switching device connected to the SVC controller dictates the resources needed for call record handling at the SVC controller. Recently switches such as carrier-class ATM backbone switches have been developed to handle in the hundred thousands to millions of connections, thus forcing SVC controllers to be designed to scale-up accordingly. For instance, if a call record is 1 KiloByte, then 1,000,000 call records would require a controller memory/storage capacity of 1 GigaByte. Such a demand for memory on an SVC leads to

increased costs in new SVCs and upgrade difficulty if existing SVCs are used with high-capacity switches.

This expense and difficulty is increased substantially because of an industry standard that demands a service availability of 99.999%. A switching node must have a service outage of no more than 3 minutes per year. To prevent against power failure/system reset, the memory used must be non-volatile (or a redundant controller unit should be hot-standby available) and further, must be protected by a Memory Management Unit (MMU) to prevent wild pointer writes and other memory failures/errors. MMU-protected memory is expensive. If a standby controller is used with a volatile memory, the bandwidth required to transfer call records from the active controller to the standby controller can be prohibitive. In either case, whether using expensive non-volatile memory or using a redundant standby controller, the providing of resources becomes critical to cost and design.

SUMMARY OF THE INVENTION

What is disclosed is a method of managing resources in a network controller connecting to a plurality of interfaces comprising recognizing a transition in the phase of a call transported through the controller, and the size of the call record of said call in accordance with the type of phase transition recognized.

Other objects, features, and advantages of the present invention will be apparent from the accompanying drawings and from the detailed description that follows below.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not limitation in the figures of the accompanying drawings in which like references indicated similar elements and in which:

Figure 1 illustrates a prior art exemplary networking topology.

Figure 2 illustrates the operation of a connection-oriented network point-to-point call with call record management.

Figure 3 illustrates the operation of a connection-oriented network point-to-multi-point call with call record management.

Figure 4 illustrates a methodology for tracking the aggregate number of calls in the establishment.

Figure 5 is a diagram of a system that can use call record management.

DETAILED DESCRIPTION

A resource management strategy is described for a network controller device such as an SVC (Switched Virtual Circuit) controller that interprets and processes call messages transported in a connection-oriented network by means of high-capacity switching devices. As will be described in greater detail below, the call record of given call is compressed or expanded depending upon the transition in phase that a call is undergoing, if any. The strategy is applicable to both point-to-point and point-to-multi-point calls

Figure 2 illustrates the operation of a connection-oriented network point-to-point call with call record management.

A "point-to-point" call is a call between two and only parties, and has three phases—establishment, active and release. For a given point-to-point call, before the call is established, the call is said to be in an idle state 200. The establishment phase 210 begins with the receipt of a Call Setup message at a particular node. The establishment phase also commences the formation of a new call record for that call as it passes that particular node. Thus, a call that has to traverse through five nodes before being connected would have five call records associated with it, one at the controller of each node. When a call is connected

(established) successfully, then that call is in the active phase 220. As such, the call is no longer "in progress," but rather has been fully established. If a call is unable to connect, then the call transitions directly back to idle state 200 from establishment phase 210. For example a destination busy signal would end the establishment phase 210 and, thus trigger the idle state before the call reaches the active phase. In this instance, the call record is fully discarded.

If a call is in the active phase 220, the transfer of user data (rather than just control/signaling data) can occur until a release (hang-up) is completed. When the call transitions from the establishment phase 210 to the active phase 220, the call record at each node that the call has passed through is updated in selected fields. Certain other fields, which the call record needed to preserve during establishment phase 210, are not needed during the active phase. According to an embodiment of the invention, the call record is compressed by discarding such information when a call reaches the active phase.

When a call completes its data transfer or is disconnected prematurely by accident or design, the active phase 220 is terminated and the call transitions to a release phase 230. The release phase 230 must be completed before the call is considered to be in the idle state 200 and ordinarily

this requires the call record to be expanded to include certain release phase information. According to one embodiment of the invention, the compressed call records from the active phase can be expanded and re-created as release phase records capable of completing the release process. When the release phase 230 is completed, the call record is discarded.

Figure 3 illustrates the operation of a connection-oriented network point-to-multi-point call with call record management.

In a "point-to-multi-point" call, two or more parties are connected together in the same active session. The point-to-multi-point call will have the ability to add or drop new parties as desired, in addition to the two parties participating in the initial call establishment ("call initiating parties"). During the active phase, no additional parties are being added or dropped from the call.

For a given point-to-multi-point call, before a call is established, the call is said to be in an idle state 300. The establishment phase 310 begins with the receipt of a Call Setup message at a particular node. The establishment phase also commences the formation of a new call record for that call as it passes that particular node. Thus, a call that has to traverse through five nodes before being connected would

have five call records associated with it, one at the controller of each node. When a call is connected (established) successfully, then that call is first in the active phase 320. As such, the call is no longer "in progress," but rather has been fully established. If a call is unable to connect, then the call transitions directly back to idle state 300 from establishment phase 310. For example a destination busy signal would end the establishment phase 310 and, thus trigger the idle state before the call reaches the active phase. In this instance, the call record is fully discarded.

If a call is in the active phase 320, the transfer of user data (rather than just control/signaling data) can occur until a release (hang-up) is completed. When the call transitions from the establishment phase 310 to the active phase 320, the call record at each node that the call has passed through is updated in selected fields. Certain other fields, which the call record needed to preserve during establishment phase 310, may not needed during the active phase. According to the invention, the call record is compressed by discarding such information when a call reaches the active phase.

In many regards, the active phase for a point-to-multi-point call is similar to the active phase for a point-to-point

call. However, unlike a point-to-point call, a point-to-multi-point call is capable of adding (and subsequently dropping) additional parties to the connection. **Figure 3** illustrates this capability by a special intermediary state, the "Add/Drop Party" state 325. When a new party is being added to the call, it is similar to a call establishment. Thus, as shown in **Figure 3**, the compressed active phase call record must be temporarily expanded when the add party procedure is initiated. When the add party procedure is completed, the call returns to the active phase, and the call record can again be compressed. A drop party procedure is similar to a call release. Thus, as shown in **Figure 3**, the compressed active phase call record must be temporarily expanded when the add party procedure is initiated. When the drop party procedure is completed, the call returns to the active phase, and the call record can again be compressed. In the Add/Drop Party state 325, the call record is expanded to include, as appropriate, information relevant to either an add or drop party proceeding.

The add party situation does not result in the creation of an entirely new call record, but adds to the existing call record. Once the add party is complete, the added party is "active" and thus, the call record can again be compressed. Likewise, when an added party is being dropped, the call

record is first expanded and after the completion of the drop party, the call record is once again compressed. A compressed call record after the completion of an add party will have more information than that of a compressed call record after the completion of a drop party.

When the call completes its data transfer or disconnected prematurely by accident or design, the active phase 320 is terminated and the call transitions to a release phase 330. The release phase 330 must be completed before the call is considered to be in the idle state 300 and ordinarily, this requires the call record to be expanded to include certain release phase information. According to one embodiment of the invention, the compressed call records from the active phase can be expanded and re-created as release phase records capable of completing the release process. When the release phase 330 is completed the call record is discarded completely.

Figure 4 illustrates fields in an exemplary call record.

A typical call record 400 is composed of fields that store particular information about a call. A unique call ID field 410 identifies a call uniquely from any other call on the node. Field 410 is critical for accessing the proper call record for a given call, and thus is maintained throughout all call phases. Also, a field 420 containing information for a

status inquiry which indicates the current state of the call (i.e., whether it is being established, already connected and so on). Field established 430 includes traffic and quality of service parameters which define the traffic flow when the call is established. Field 440 includes addressing and routing information in order to trace the path of the call for diagnosis, or to identify where in the network a message for that should be forwarded. A field 450 contains call accounting information, such as the call length or time/date the call was established. Field 460 contains timer information used to determine if a time-out situation has occurred. Field 470 contains retry counters to determine how many times a call set-up retry should be attempted. Field 490 contains pointers to setup messages that are being processed or forwarded by the controller.

According to one embodiment of the invention, fields 460, 470, and 490 may be discarded as soon as a call enters the active phase. The completion of the establishment phase eliminates the need for the information in those fields. By freeing the allocated memory for those fields (rather than merely clearing the fields to null values) extra memory may be made available for other call record storage or other systemic use. By compressing these fields, the call record is compressed. Point-to-point and point-to-multi-point have

similar call record structures, with the exception of fields related to added parties dynamically created in the point-to-multi-point call. Such fields include pointers to mini-call records ("child" records of a "root" such as call record 400), which can be added and discarded as a party is added or dropped. When call records are expanded, memory is allocated for fields that are needed for the phase sought to be completed.

Figure 5 is a diagram of a system that can use call record management.

A network node in a connection-oriented network, such as the nodes shown in **Figure 1**, incorporates at least two elements--an active SVC controller 500 and a switch device 510. Such a node may also include a standby SVC controller 520 that takes substitute control of the system when active controller 500 fails to operate as expected. Switch 510 connects to active controller 500 over a number of bi-directional interfaces which pass through switch 510 connecting to either other nodes on the network or to user terminals under the purview of the node in which the switch 510 and controller 500 function. Controller 500 accepts messages over these interfaces which belong to calls that pass through the node. These messages are interpreted and processed by a message processing system 507, which may itself

incorporate processors, buffers, protocol stacks and signaling mechanisms, which then initiates action based on the content and directives, if any, contained therein. For instance, when a call setup is successfully processed, a new call record may be created in a call record memory 505 (which may be physically distinct or coalesced with other memories in controller 500 or external to the controller itself (not pictured)). Call record memory 505 is illustrated as containing N call records but this number may be increased as further call setup messages are encountered. Each call record, according to one or more embodiments of the invention, is either in compressed or expanded form, depending upon the phase which the call is entering (or exiting). For instance, when a message indicating a call connect is processed by message processing system 507, controller 500 will compress the call record indicated in the call connect message (by its unique call ID) by removing fields pertinent only to the establishment phase. Likewise, when a disconnect message is processed, the call record is expanded by controller 500 to include release phase related fields. Further, call records can dynamically point to other data structures that store information regarding added or dropped parties in a point-to-multi-point call. As a result of compressing call records and expanding them only as needed, the average size of a call

record during its lifetime can be reduced, and thus, the amount of memory restricted for call records can be minimized. Further, in node where a standby controller 520 is used, as in **Figure 5**, the amount of bandwidth needed to transfer over call records in the event of a failure would be lower and have the intended advantage of making the process of transferring such records less time consuming, thus allowing the standby controller 520 to go active more rapidly.

In the foregoing specification, the invention has been described with reference to specific exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.